Extensible Messaging and Presence Protocol (XMPP): Address Format
draft-ietf-xmpp-address-05

Abstract

This document defines the format for addresses used in the Extensible Messaging and Presence Protocol (XMPP), including support for non-ASCII characters.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 9, 2011.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.
# Table of Contents

1. Introduction ............................................. 3
2. Addresses .................................................. 3
   2.1. Fundamentals ........................................... 4
   2.2. Domainpart ............................................ 5
   2.3. Localpart .............................................. 6
   2.4. Resourcepart .......................................... 7
3. Internationalization Considerations .......................... 8
4. Security Considerations ................................. 8
   4.1. Reuse of Stringprep .................................. 8
   4.2. Reuse of Unicode .................................... 8
   4.3. Confusable Characters ............................... 8
   4.4. Address Spoofing .................................... 9
      4.4.1. Address Forging .................................. 9
      4.4.2. Address Mimicking ............................... 10
5. IANA Considerations ..................................... 11
   5.1. Nodeprep Profile of Stringprep ...................... 11
   5.2. Resourceprep Profile of Stringprep ................ 11
6. Conformance Requirements ................................ 12
7. References ................................................. 13
   7.1. Normative References ................................ 13
   7.2. Informative References ............................. 14
Appendix A. Nodeprep ........................................ 16
   A.1. Introduction .......................................... 16
   A.2. Character Repertoire ................................ 16
   A.3. Mapping .............................................. 17
   A.4. Normalization ....................................... 17
   A.5. Prohibited Output .................................... 17
   A.6. Bidirectional Characters ........................... 17
   A.7. Notes ................................................. 18
Appendix B. Resourceprep ..................................... 18
   B.1. Introduction .......................................... 18
   B.2. Character Repertoire ................................ 18
   B.3. Mapping .............................................. 19
   B.4. Normalization ....................................... 19
   B.5. Prohibited Output .................................... 19
   B.6. Bidirectional Characters ........................... 19
Appendix C. Differences From RFC 3920 ...................... 20
Author’s Address ............................................ 20
1. Introduction

The Extensible Messaging and Presence Protocol (XMPP) is an application profile of the Extensible Markup Language [XML] for streaming XML data in close to real time between any two or more network-aware entities. The address format for XMPP entities was originally developed in the Jabber open-source community in 1999, first described by [XEP-0029] in 2002, and defined canonically by [RFC3920] in 2004.

As specified in RFC 3920, the XMPP address format re-uses the "stringprep" technology for preparation of non-ASCII characters [STRINGPREP], including the Nameprep profile for internationalized domain names as specified in [NAMEPREP] and [IDNA2003] along with two XMPP-specific profiles for the localpart and resourcepart.

Since the publication of RFC 3920, IDNA2003 has been superseded by IDNA2008 (see [IDNA-PROTO] and related documents), which is not based on stringprep. Following the lead of the IDNA community, other technology communities that use stringprep have begun discussions about migrating away from stringprep toward more "modern" approaches. The XMPP community is participating in those discussions in order to find a replacement for the Nodeprep and Resourceprep profiles of stringprep defined in RFC 3920. However, work on improved handling of internationalized addresses is currently in progress within the PRECIS Working Group and at the time of this writing it seems that such work might take several years to complete. Because all other aspects of revised documentation for XMPP have been incorporated into [rfc3920bis], the XMPP Working Group decided to split the XMPP address format into a separate specification so as not to significantly delay publication of improved documentation for XMPP while awaiting the conclusion of work on improved handling of internationalized addresses.

Therefore, this specification provides corrected documentation of the XMPP address format using the internationalization technologies available in 2004 (when RFC 3920 was published), with the intent that this specification will be superseded as soon as work on a new approach to preparation and comparison of internationalized strings has been defined by the PRECIS Working Group and applied to the specific cases of XMPP localparts and resourceparts.

2. Addresses
2.1. Fundamentals

An XMPP entity is anything that is network-addressable and that can communicate using XMPP. For historical reasons, the native address of an XMPP entity is called a Jabber Identifier or JID. A valid JID is a string of [UNICODE] code points, encoded using [UTF-8], and structured as an ordered sequence of localpart, domainpart, and resourcepart (where the first two parts are demarcated by the '@' character used as a separator, and the last two parts are similarly demarcated by the '/' character).

The syntax for a JID is defined as follows using the Augmented Backus-Naur Form as specified in [ABNF].

```
  jid           = [ localpart "@" ] domainpart [ "/" resourcepart ]
  localpart     = 1*(nodepoint)
    ; a "nodepoint" is a UTF-8 encoded Unicode code point that satisfies the Nodeprep profile of stringprep
  domainpart    = IP-literal / IPv4address / ifqdn
    ; the "IPv4address" and "IP-literal" rules are defined in RFC 3986, and the first-match-wins (a.k.a. "greedy") algorithm described in RFC 3986 applies to the matching process
    ; ifqdn         = 1*(namepoint)
    ; a "namepoint" is a UTF-8 encoded Unicode code point that satisfies the Nameprep profile of stringprep
  ifqdn         = 1*(resourcepoint)
    ; a "resourcepoint" is a UTF-8 encoded Unicode code point that satisfies the Resourceprep profile of stringprep
  resourcepart  = 1*(resourcepoint)
```

All JIDs are based on the foregoing structure. One common use of this structure is to identify a messaging and presence account, the server that hosts the account, and a connected resource (e.g., a specific device) in the form of <localpart@domain/resource>. However, localparts other than clients are possible; for example, a specific chat room offered by a multi-user chat service (see [XEP-0045]) is addressed as <room@service> (where "room" is the name of the chat room and "service" is the hostname of the multi-user chat service) and a specific occupant of such a room could be addressed as <room@service/nick> (where "nick" is the occupant’s room nickname). Many other JID types are possible (e.g., <domain/resource> could be a server-side script or service).

Each allowable portion of a JID (localpart, domainpart, and resourcepart) MUST NOT be zero bytes in length and MUST NOT be more
than 1023 bytes in length, resulting in a maximum total size (including the ‘@’ and ‘/’ separators) of 3071 bytes.

An entity’s address on an XMPP network MUST be represented as a JID (without a URI scheme) and not a [URI] or [IRI] as specified in [XMPP-URI]; the latter specification is provided only for identification and interaction outside the context of XMPP itself.

Implementation Note: When dividing a JID into its component parts, an implementation needs to match the separator characters ‘@’ and ‘/’ before applying any transformation algorithms, which might decompose certain Unicode code points to the separator characters (e.g., U+FE6B SMALL COMMERCIAL AT might decompose into U+0040 COMMERCIAL AT).

2.2. Domainpart

The DOMAINPART of a JID is that portion after the ‘@’ character (if any) and before the ‘/’ character (if any); it is the primary identifier and is the only REQUIRED element of a JID (a mere domainpart is a valid JID). Typically a domainpart identifies the "home" server to which clients connect for XML routing and data management functionality. However, it is not necessary for an XMPP domainpart to identify an entity that provides core XMPP server functionality (e.g., a domainpart can identify an entity such as a multi-user chat service, a publish-subscribe service, or a user directory).

The domainpart for every server or service that will communicate over a network SHOULD be a fully qualified domain name or "FQDN" (see [DNS]); although the domainpart is allowed to be either an Internet Protocol (IPv4 or IPv6) address or a text label that is resolvable on a local network (commonly called an "unqualified hostname"), it is possible that domainparts that are IP addresses will not be acceptable to other services for the sake of interdomain communication. Furthermore, domainparts that are unqualified hostnames MUST NOT be used on public networks but MAY be used on private networks.

Note: If the domainpart includes a final character considered to be a label separator (dot) by [IDNA2003] or [DNS], this character MUST be stripped from the domainpart before the JID of which it is a part is used for the purpose of routing an XML stanza, comparing against another JID, or constructing an [XMPP-URI]; in particular, the character MUST be stripped before any other canonicalization steps are taken, such as application of the [NAMEPREP] profile of [STRINGPREP] or completion of the ToASCII operation as described in [IDNA2003].
A domainpart MUST NOT be zero bytes in length and MUST NOT be more than 1023 bytes in length.

A domainpart consisting of a fully qualified domain name MUST be an "internationalized domain name" as defined in [IDNA2003], that is, it MUST be "a domain name in which every label is an internationalized label" and MUST follow the rules for construction of internationalized domain names specified in [IDNA2003]. When preparing a text label (consisting of a sequence of UTF-8 encoded Unicode code points) for representation as an internationalized label in the process of constructing an XMPP domainpart or comparing two XMPP domainparts, an application MUST ensure that for each text label it is possible to apply without failing the ToASCII operation specified in [IDNA2003] with the UseSTD3ASCIIRules flag set (thus forbidding ASCII code points other than letters, digits, and hyphens). If the ToASCII operation can be applied without failing, then the label is an internationalized label. (Note: The ToASCII operation includes application of the [NAMEPREP] profile of [STRINGPREP] and encoding using the algorithm specified in [PUNYCODE]; for details, see [IDNA2003].) Although XMPP applications do not communicate the output of the ToASCII operation (called an "ACE label") over the wire, it MUST be possible to apply that operation without failing to each internationalized label. If an XMPP application receives as input an ACE label, it SHOULD convert that ACE label to an internationalized label using the ToUnicode operation (see [IDNA2003]) before including the label in an XMPP domainpart that will be communicated over the wire on an XMPP network (however, instead of converting the label, there are legitimate reasons why an application might instead refuse the input altogether and return an error to the entity that provided the offending data).

In the terms of IDNA2008 [IDNA-DEFS], the domainpart of a JID is a "domain name slot".

2.3. Localpart

The LOCALPART of a JID is an optional identifier placed before the domainpart and separated from the latter by the '@' character. Typically a localpart uniquely identifies the entity requesting and using network access provided by a server (i.e., a local account), although it can also represent other kinds of entities (e.g., a chat room associated with a multi-user chat service). The entity represented by an XMPP localpart is addressed within the context of a specific domain.

A localpart MUST NOT be zero bytes in length and MUST NOT be more than 1023 bytes in length.
A localpart MUST be formatted such that the Nodeprep profile of [STRINGPREP] can be applied without failing (see Appendix A). Before comparing two localparts, an application MUST first ensure that the Nodeprep profile has been applied to each identifier (the profile need not be applied each time a comparison is made, as long as it has been applied before comparison).

2.4. Resourcepart

The resourcepart of a JID is an optional identifier placed after the domainpart and separated from the latter by the '/-' character. A resourcepart can modify either a <localpart@domainpart> address or a mere <domainpart> address. Typically a resourcepart uniquely identifies a specific connection (e.g., a device or location) or object (e.g., an occupant in a multi-user chat room) belonging to the entity associated with an XMPP localpart at a local domain.

When an XMPP address does not include a resourcepart (i.e., when it is of the form <domainpart> or <localpart@domainpart>), it is referred to as a BARE JID. When an XMPP address includes a resourcepart (i.e., when it is of the form <domain/resource> or <localpart@domain/resource>), is referred to as a FULL JID.

A resourcepart MUST NOT be zero bytes in length and MUST NOT be more than 1023 bytes in length.

A resourcepart MUST be formatted such that the Resourceprep profile of [STRINGPREP] can be applied without failing (see Appendix B). Before comparing two resourceparts, an application MUST first ensure that the Resourceprep profile has been applied to each identifier (the profile need not be applied each time a comparison is made, as long as it has been applied before comparison).

Note: For historical reasons, the term "resource identifier" is often used in XMPP to refer to the optional portion of an XMPP address that follows the domainpart and the "/'" separator character; to help prevent confusion between an XMPP "resource identifier" and the meanings of "resource" and "identifier" provided in Section 1.1 of [URI], this specification uses the term "resourcepart" instead of "resource identifier" (as in RFC 3920).

XMPP entities SHOULD consider resourceparts to be opaque strings and SHOULD NOT impute meaning to any given resourcepart. In particular:

- Use of the '/-' character as a separator between the domainpart and the resourcepart does not imply that XMPP addresses are hierarchical in the way that, say, HTTP addresses are hierarchical; thus for example an XMPP address of the form
<localpart@domain/foo/bar> does not identify a resource "bar" that
exists below a resource "foo" in a hierarchy of resources
associated with the entity "localpart@domain".

- The '@' character is allowed in the resourcepart, and is often
  used in the "nick" shown in XMPP chatrooms. For example, the JID
  <room@chat.example.com/user@host> describes an entity who is an
  occupant of the room <room@chat.example.com> with an (asserted)
  nick of <user@host>. However, chatroom services do not
  necessarily check such an asserted nick against the occupant’s
  real JID.

3. Internationalization Considerations

XMPP servers MUST, and XMPP clients SHOULD, support [IDNA2003] for
domainparts (including the [NAMEPREP] profile of [STRINGPREP]), the
Nodeprep (Appendix A) profile of [STRINGPREP] for localparts, and the
Resourceprep (Appendix B) profile of [STRINGPREP] for resourceparts;
this enables XMPP addresses to include a wide variety of characters
outside the US-ASCII range. Rules for enforcement of the XMPP
address format are provided in [rfc3920bis].

4. Security Considerations

4.1. Reuse of Stringprep

The security considerations described in [STRINGPREP] apply to the
Nodeprep (Appendix A) and Resourceprep (Appendix B) profiles defined
in this document for XMPP localparts and resourceparts. The security
considerations described in [STRINGPREP] and [NAMEPREP] apply to the
Nameprep profile that is re-used here for XMPP domainparts.

4.2. Reuse of Unicode

The security considerations described in [UNICODE-SEC] apply to the
use of Unicode characters in XMPP addresses.

4.3. Confusable Characters

The Unicode and ISO/IEC 10646 repertoires have many characters that
look similar (so-called "confusable characters"). In many cases,
users of security protocols might perform visual matching, such as
when comparing the names of trusted third parties. Because it is
impossible to map similar-looking characters without a great deal of
context (such as knowing the fonts used), stringprep does nothing to
map similar-looking characters together, nor to prohibit some
characters because they look like others. Some specific suggestions about identification and handling of confusable characters appear in the Unicode Security Considerations [UNICODE-SEC].

A localpart can be employed as one part of an entity’s address in XMPP. One common usage is as the username of an instant messaging user; another is as the name of a multi-user chat room; and many other kinds of entities could use localparts as part of their addresses. The security of such services could be compromised based on different interpretations of the internationalized localpart; for example, a user entering a single internationalized localpart could access another user’s account information, or a user could gain access to a hidden or otherwise restricted chat room or service.

A resourcepart can be employed as one part of an entity’s address in XMPP. One common usage is as the name for an instant messaging user’s connected resource; another is as the nickname of a user in a multi-user chat room; and many other kinds of entities could use resourceparts as part of their addresses. The security of such services could be compromised based on different interpretations of the internationalized resourcepart; for example, a user could attempt to initiate multiple connections with the same name, or a user could send a message to someone other than the intended recipient in a multi-user chat room.

4.4. Address Spoofing

There are two forms of address spoofing: forging and mimicking.

4.4.1. Address Forging

In the context of XMPP technologies, address forging occurs when an entity is able to generate an XML stanza whose ‘from’ address does not correspond to the account credentials with which the entity authenticated onto the network (or an authorization identity provided during SASL negotiation). For example, address forging occurs if an entity that authenticated as "juliet@im.example.com" is able to send XML stanzas from "nurse@im.example.com" or "romeo@example.net".

Address forging is difficult in XMPP systems, given the requirement for sending servers to stamp ‘from’ addresses and for receiving servers to verify sending domains via server-to-server authentication (see [rfc3920bis]). However, address forging is not impossible, since a rogue server could forge JIDs at the sending domain by ignoring the stamping requirement. Therefore, an entity outside the security perimeter of a particular server cannot reliably distinguish between bare JIDs of the form <localpart@domainpart> at that server and thus can authenticate only the domainpart of such JIDs with any
level of assurance. This specification does not define methods for
discovering or counteracting such rogue servers.

Furthermore, it is possible for an attacker to forge JIDs at other
domains by means of a DNS poisoning attack if DNS security extensions
[DNSSEC] are not used.

4.4.2. Address Mimicking

Address mimicking occurs when an entity provides legitimate
authentication credentials for and sends XML stanzas from an account
whose JID appears to a human user to be the same as another JID. For
example, in some XMPP clients the address "paypal@example.org"
(spelled with the number one as the final character of the localpart)
might appear to be the same as "paypal@example.org" (spelled with the
lower-case version of the letter "l"), especially on casual visual
inspection; this phenomenon is sometimes called "typejacking". A
more sophisticated example of address mimicking might involve the use
of characters from outside the US-ASCII range, such as the Cherokee
characters U+13DA U+13A2 U+13B5 U+13AC U+13A2 U+13AC U+13D2 instead
of the US-ASCII characters "STPETER".

In some examples of address mimicking, it is unlikely that the
average user could tell the difference between the real JID and the
fake JID. (Indeed, there is no way to distinguish with full
certainty which is the fake JID and which is the real JID; in some
communication contexts, the JID with Cherokee characters might be the
real JID and the JID with US-ASCII characters might thus appear to be
the fake JID.) Because JIDs can contain almost any Unicode
character, it can be relatively easy to mimic some JIDs in XMPP
systems. The possibility of address mimicking introduces security
vulnerabilities of the kind that have also plagued the World Wide
Web, specifically the phenomenon known as phishing.

As noted in [IDNA-DEFS], "there are no comprehensive technical
solutions to the problems of confusable characters". Mimicked JIDs
that involve characters from only one character set or from the
character set typically employed by a particular user are not easy to
combat (e.g., the simple typejacking attack previously described,
which relies on a surface similarity between the characters "1" and
"l" in some presentations). However, mimicked addresses that involve
characters from more than one character set, or from a character set
not typically employed by a particular user, can be mitigated
somewhat through intelligent presentation. In particular, every
human user of an XMPP technology presumably has a preferred language
(or, in some cases, a small set of preferred languages), which an
XMPP application SHOULD gather either explicitly from the user or
implicitly via the operating system of the user’s device.
Furthermore, every language has a range (or a small set of ranges) of characters normally used to represent that language in textual form. Therefore, an XMPP application SHOULD warn the user when presenting a JID that mixes characters from more than one character set or that uses characters outside the normal range of the user’s preferred language(s). This recommendation is not intended to discourage communication across language communities; instead, it recognizes the existence of such language communities and encourages due caution when presenting unfamiliar character sets to human users.

5. IANA Considerations

   The following sections update the registrations provided in [RFC3920].

5.1. Nodeprep Profile of Stringprep

   The Nodeprep profile of stringprep is defined under Nodeprep (Appendix A). The IANA has registered Nodeprep in the stringprep profile registry.

   Name of this profile:

       Nodeprep

   RFC in which the profile is defined:

       XXXX

   Indicator whether or not this is the newest version of the profile:

       This is the first version of Nodeprep

5.2. Resourceprep Profile of Stringprep

   The Resourceprep profile of stringprep is defined under Resourceprep (Appendix B). The IANA has registered Resourceprep in the stringprep profile registry.

   Name of this profile:

       Resourceprep

   RFC in which the profile is defined:
6. Conformance Requirements

This section describes a protocol feature set that summarizes the conformance requirements of this specification. This feature set is appropriate for use in software certification, interoperability testing, and implementation reports. For each feature, this section provides the following information:

- A human-readable name
- An informational description
- A reference to the particular section of this document that normatively defines the feature
- Whether the feature applies to the Client role, the Server role, or both (where "N/A" signifies that the feature is not applicable to the specified role)
- Whether the feature MUST or SHOULD be implemented, where the capitalized terms are to be understood as described in [KEYWORDS]

The feature set specified here attempts to adhere to the concepts and formats proposed by Larry Masinter within the IETF’s NEWTRK Working Group in 2005, as captured in [INTEROP]. Although this feature set is more detailed than called for by [REPORTS], it provides a suitable basis for the generation of implementation reports to be submitted in support of advancing this specification from Proposed Standard to Draft Standard in accordance with [PROCESS].

Feature: address-domain-length
Description: Ensure that the domainpart of an XMPP address is at least one byte in length and at most 1023 bytes in length.
Section: Section 2.2
Roles: Both MUST.

Feature: address-domain-prep
Description: Ensure that the domainpart of an XMPP address conforms to the Nameprep profile of Stringprep.
Section: Section 2.2
Roles: Client SHOULD, Server MUST.

Feature: address-localpart-length
Description: Ensure that the localpart of an XMPP address is at least one byte in length and at most 1023 bytes in length.
Section: Section 2.3
Roles: Both MUST.

Feature: address-localpart-prep
Description: Ensure that the localpart of an XMPP address conforms to the Nodeprep profile of Stringprep.
Section: Section 2.3
Roles: Client SHOULD, Server MUST.

Feature: address-resource-length
Description: Ensure that the resourcepart of an XMPP address is at least one byte in length and at most 1023 bytes in length.
Section: Section 2.4
Roles: Both MUST.

Feature: address-resource-prep
Description: Ensure that the resourcepart of an XMPP address conforms to the Resourceprep profile of Stringprep.
Section: Section 2.2
Roles: Client SHOULD, Server MUST.

7. References

7.1. Normative References


See Section 1 for an explanation of why the normative reference to an obsoleted specification is needed.

[NAMEPREP]

See Section 1 for an explanation of why the normative reference to an obsoleted specification is needed.

/rfc3920bis

/STRINGPREP

/UNICODE


/UNICODE-SEC

=UTF-8

7.2. Informative References

/DNS

/DNSSEC

[IDNA-DEFS]
Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework",
RFC 5890, August 2010.


APPENDIX A.  NODEPREP

A.1.  Introduction

This appendix defines the "Nodeprep" profile of stringprep.  As such, it specifies processing rules that will enable users to enter internationalized localparts in the Extensible Messaging and Presence Protocol (XMPP) and have the highest chance of getting the content of the strings correct.  (An XMPP localpart is the optional portion of an XMPP address that precedes an XMPP domainpart and the '@' separator; it is often but not exclusively associated with an instant messaging username.)  These processing rules are intended only for XMPP localparts and are not intended for arbitrary text or any other aspect of an XMPP address.

This profile defines the following, as required by [STRINGPREP]:

- The intended applicability of the profile: internationalized localparts within XMPP
- The character repertoire that is the input and output to stringprep: Unicode 3.2, specified in Section 2 of this Appendix
- The mappings used: specified in Section 3
- The Unicode normalization used: specified in Section 4
- The characters that are prohibited as output: specified in Section 5
- Bidirectional character handling: specified in Section 6

A.2.  Character Repertoire

This profile uses Unicode 3.2 with the list of unassigned code points being Table A.1, both defined in Appendix A of [STRINGPREP].
A.3. Mapping

This profile specifies mapping using the following tables from [STRINGPREP]:

Table B.1
Table B.2

A.4. Normalization

This profile specifies the use of Unicode normalization form KC, as described in [STRINGPREP].

A.5. Prohibited Output

This profile specifies the prohibition of using the following tables from [STRINGPREP].

Table C.1.1
Table C.1.2
Table C.2.1
Table C.2.2
Table C.3
Table C.4
Table C.5
Table C.6
Table C.7
Table C.8
Table C.9

In addition, the following additional Unicode characters are also prohibited:

U+0022 (QUOTATION MARK), i.e., "
U+0026 (AMPERSAND), i.e., &
U+0027 (APOSTROPHE), i.e., '
U+002F (SOLIDUS), i.e., /
U+003A (COLON), i.e., :
U+003C (LESS-THAN SIGN), i.e., <
U+003E (GREATER-THAN SIGN), i.e., >
U+0040 (COMMERCIAL AT), i.e., @

A.6. Bidirectional Characters

This profile specifies checking bidirectional strings, as described in Section 6 of [STRINGPREP].
A.7. Notes

Because the additional characters prohibited by Nodeprep are prohibited after normalization, an implementation MUST NOT enable a human user to input any Unicode code point whose decomposition includes those characters; such code points include but are not necessarily limited to the following (refer to [UNICODE] for complete information).

- U+2100 (ACCOUNT OF)
- U+2101 (ADDRESSED TO THE SUBJECT)
- U+2105 (CARE OF)
- U+2106 (CADA UNA)
- U+226E (NOT LESS-THAN)
- U+226F (NOT GREATER-THAN)
- U+2A74 (DOUBLE COLON EQUAL)
- U+FE13 (SMALL COLON)
- U+FE60 (SMALL AMPERSAND)
- U+FE64 (SMALL LESS-THEAN SIGN)
- U+FE65 (SMALL GREATER-THEAN SIGN)
- U+FE6B (SMALL COMMERCIAL AT)
- U+FF02 (FULLWIDTH QUOTATION MARK)
- U+FF06 (FULLWIDTH AMPERSAND)
- U+FF07 (FULLWIDTH APOSTROPHE)
- U+FF0F (FULLWIDTH SOLIDUS)
- U+FF1A (FULLWIDTH COLON)
- U+FF1C (FULLWIDTH LESS-THEAN SIGN)
- U+FF1E (FULLWIDTH GREATER-THEAN SIGN)
- U+FF20 (FULLWIDTH COMMERCIAL AT)

Appendix B. Resourceprep

B.1. Introduction

This appendix defines the "Resourceprep" profile of stringprep. As such, it specifies processing rules that will enable users to enter internationalized resourceparts in the Extensible Messaging and Presence Protocol (XMPP) and have the highest chance of getting the content of the strings correct. (An XMPP resourcepart is the optional portion of an XMPP address that follows an XMPP domainpart and the '/' separator.) These processing rules are intended only for XMPP resourceparts and are not intended for arbitrary text or any other aspect of an XMPP address.

This profile defines the following, as required by [STRINGPREP]:

[Page 18]
The intended applicability of the profile: internationalized resource parts within XMPP

- The character repertoire that is the input and output to stringprep: Unicode 3.2, specified in Section 2 of this Appendix
- The mappings used: specified in Section 3
- The Unicode normalization used: specified in Section 4
- The characters that are prohibited as output: specified in Section 5
- Bidirectional character handling: specified in Section 6

B.2. Character Repertoire

This profile uses Unicode 3.2 with the list of unassigned code points being Table A.1, both defined in Appendix A of [STRINGPREP].

B.3. Mapping

This profile specifies mapping using the following tables from [STRINGPREP]:

Table B.1

B.4. Normalization

This profile specifies the use of Unicode normalization form KC, as described in [STRINGPREP].

B.5. Prohibited Output

This profile specifies the prohibition of using the following tables from [STRINGPREP].

Table C.1.2
Table C.2.1
Table C.2.2
Table C.3
Table C.4
Table C.5
Table C.6
Table C.7
Table C.8
Table C.9

B.6. Bidirectional Characters

This profile specifies checking bidirectional strings, as described in Section 6 of [STRINGPREP].
Appendix C. Differences From RFC 3920

Based on consensus derived from implementation and deployment experience as well as formal interoperability testing, the following substantive modifications were made from RFC 3920.

- Corrected the ABNF syntax to (1) ensure consistency with [URI] and [IRI], and (2) prevent zero-length localparts, domainparts, and resourceparts.
- To avoid confusion with the term "node" as used in [XEP-0030] and [XEP-0060], changed the term "node identifier" to "localpart" (but retained the name "Nodeprep" for backward compatibility).
- To avoid confusion with the terms "resource" and "identifier" as used in [URI], changed the term "resource identifier" to "resourcepart".
- Corrected the nameprep processing rules to require use of the UseSTD3ASCIIRules flag.

Author’s Address

Peter Saint-Andre
Cisco
1899 Wyknoop Street, Suite 600
Denver, CO  80202
USA

Phone: +1-303-308-3282
Email: psaintan@cisco.com